

PAPER • OPEN ACCESS

## Using PERT and Gantt charts for planning software projects on the basis of distributed digital ecosystems

To cite this article: Ivan Valeryevich Evdokimov *et al* 2018 *J. Phys.: Conf. Ser.* **1074** 012127

View the [article online](#) for updates and enhancements.



**IOP | ebooks™**

Bringing you innovative digital publishing with leading voices to create your essential collection of books in STEM research.

Start exploring the **collection** - download the first chapter of every title for free.

# Using PERT and Gantt charts for planning software projects on the basis of distributed digital ecosystems

Ivan Valeryevich Evdokimov, Roman Yurievich Tsarev<sup>1</sup>, Tatiana Nikolaevna Yamskikh and Alexander Nikolaevich Pupkov

Siberian Federal University, 79 Svobodny Prospekt, Krasnoyarsk, Russia

<sup>1</sup> E-mail: tsarev.sfu@mail.ru

**Abstract.** In modern digital economy transition to a decentralized Internet allows companies to grow into a multi-billion dollar business on the basis of the simple idea to provide additional services based on distributed technology. The authors of this article believe that transition to concepts of Web 3.0, such as user-generated content, has a significant impact on industrial production and application of software based on distributed digital ecosystems. The definition of the digital distributed ecosystems which takes into account the features of the program engineering caused by the intangibility of the software product is suggested. The principles of managing large software projects with PERT and Gantt charts are considered. The simplified example of managing software project "Procurement Management" which illustrates the author's approach is provided. The reviewed example shows application of PERT chart for project management with critical path calculation. The practice of creating a Gantt chart on the basis of this project is also considered. Such approach is encouraged for large projects and it can be useful for project management at a high level of abstraction, irrespective of the project size.

## 1. Introduction

Effective management is essential for successful software system engineering [1, 2]. Software engineering management includes providing comfortable work environment for the developers, development planning and quality control [3-7].

A variety of models and methods are used to plan the process of development [8]. To analyze both early and late start and finish dates for the implementation of the project, the methods of network planning are used [9]. These methods integrate all the stages of the project, thus it is possible to determine the total duration [10].

There are both probabilistic and deterministic network-based methods. The probabilistic methods are statistical tests, the Monte Carlo simulations, Programme Evaluation and Review Technique (hereinafter referred to as PERT), which belong to the group of non-alternative [11]. As an alternative, the Graphical Evaluation Review Technique (hereinafter referred to as GERT) is distinguished. The Gantt chart and the critical path method can be classified as the deterministic ones [12].

Large projects are characterized by the complexity of constructing a plan or scheduling the project in distinct work stages. One of the methods that can be used in this case is the GERT method, based on GERT networks. With this method it is possible to determine the expected duration of the project on the basis of three probabilistic time estimates. The network-based model is a probabilistic network, taking into account the possibility to use different structures of project tasks. This allows to prevent

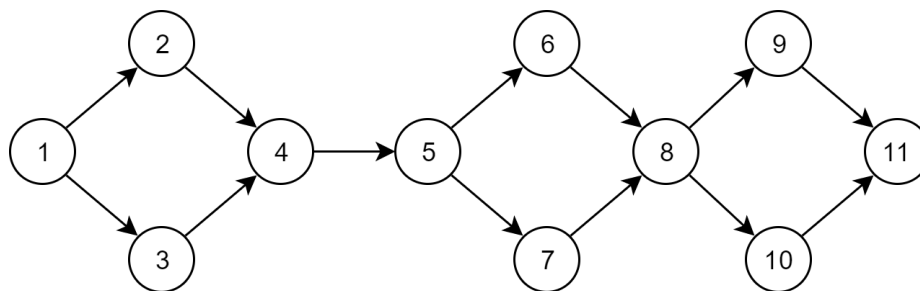


the risks during the whole project lifecycle, not only at its separate stage. However, there is one necessary condition - each stage builds upon the successful completion of earlier stages.

As a technique for schedule planning and control PERT is focused on analyzing the tasks necessary for the project implementation. The analysis of the execution time of each individual task, as well as the construction of the network schedule (PERT network diagram), considered as an independent method, make the basis of this technique. It was used to create the software system "Procurement Management".

## 2. Network planning of software development process

The network diagram is shown in figure 1. Each vertex corresponds to the single task. The arrows are used to show which tasks should be completed to start this or that stage.



**Figure 1.** Network diagram.

The stages of project development marked with numbers are indicated in table 1.

**Table 1.** Project development stages.

no.	Task title	Duration (days)
1	Beginning of the project implementation	0
2	Meeting with a customer	2
3	Investigating the subject area	6
4	Designing class models and basic algorithms	5
5	Developing the modules of the software system	10
6	Module Testing	8
7	Interface design and development	3
8	Testing and Debugging	5
9	Presenting software to a customer	1
10	Writing software documentation	3
11	Project Completion	0

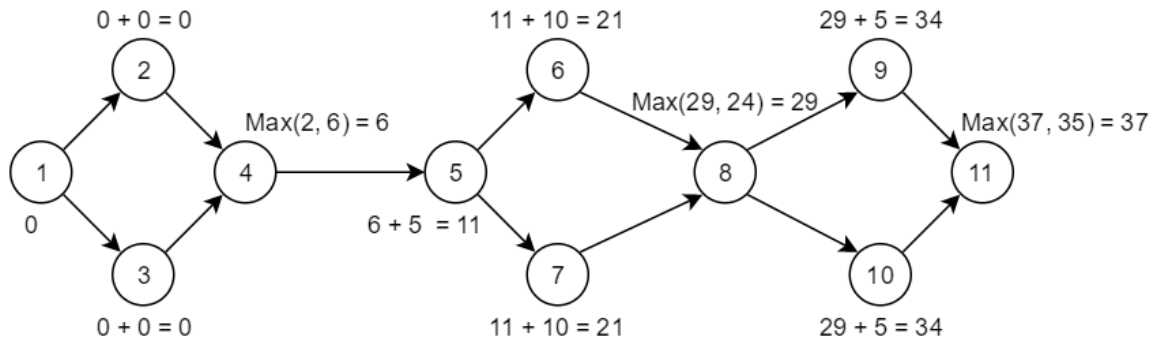
When developing both small and large projects, it is important to take into account the shortest possible time period in which the project can be completed. In this case, a critical path is calculated [13]. The method aims to determine the longest path of activities from start of the project until the end. When finding the minimum time frame for project development, it is assumed that critical path tasks do not have associated float, do not have reserve schedule time to use for delays. This condition is necessary as project development is a greatly influenced process. Wrong project scheduling, i.e. the wrong sequence of the most time-consuming tasks, will change the actual completion date of the project.

At the same time, there are tasks that are not on the critical path, but still determine the project's success. These tasks may have slack and it is possible to define two dates: the start and the finish of the stage, respectively, the earliest and latest.

Using the network diagram, we can calculate the early and late start dates for the tasks.

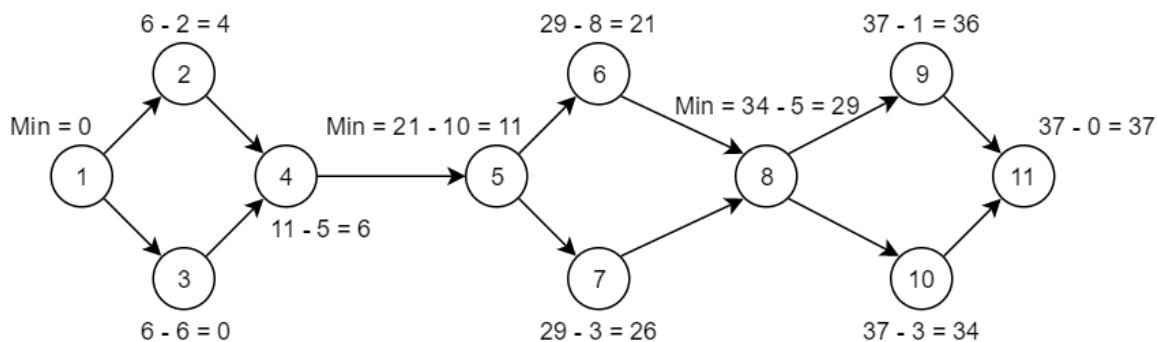
To determine the earliest start date for a stage, we must define the length of the longest path to this stage on the network diagram. Early start of the subsequent task is determined by the early finish of the previous task.

The early start and finish are determined for all chart tasks sequentially from the initial event. The greatest values of task duration are taken to calculate the early finish dates (see figure 2).



**Figure 2.** Early start date calculation.

Late start date for the task, which does not delay the development process, is calculated as the difference between the duration of the critical path and the longest path from the previous task to the final one. Late start date for a particular stage is defined as the difference between the lengths of the critical path and the longest path from the previous stage to the final one. If there are several previous stages, then a minimum value should be defined to avoid delaying subsequent tasks (see figure 3).



**Figure 3.** Early start date calculation.

The slack shows where there is flexibility in the schedule. This is an amount of time that a task can be delayed without causing a delay to subsequent tasks or project completion date. It is defined by subtracting the early start date of the stage from the late one. The calculation of slack for the project stages is presented in table 2.

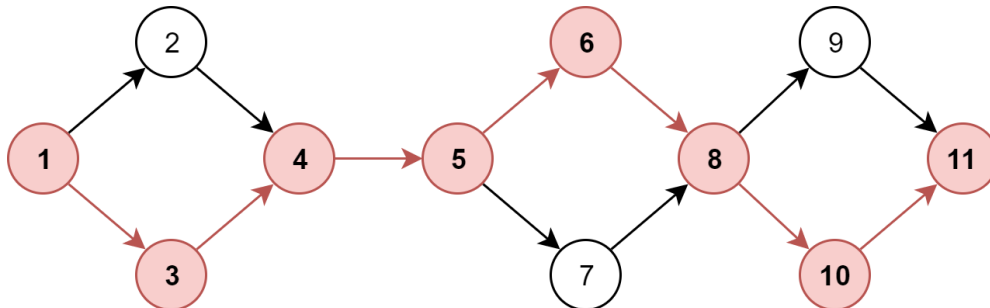
**Table 2.** Slack calculation.

Task	1	2	3	4	5	6	7	8	9	10	11
Early start date	0	0	0	6	11	21	21	29	34	34	37
Late start date	0	4	0	6	11	21	26	29	36	34	37
Slack	0	4	0	0	0	0	5	0	2	0	0

As a result, we will see the stages of development which have some slack or float. The stages with their early start date equal to their late start date have zero slack.

The sequence of project tasks which add up to the longest overall duration is called a critical path. This determines the shortest time possible to complete the project. To make sure the project finishes

on time, tasks on the critical path should be completed as soon as possible. The critical path for a network diagram is shown in figure 4.

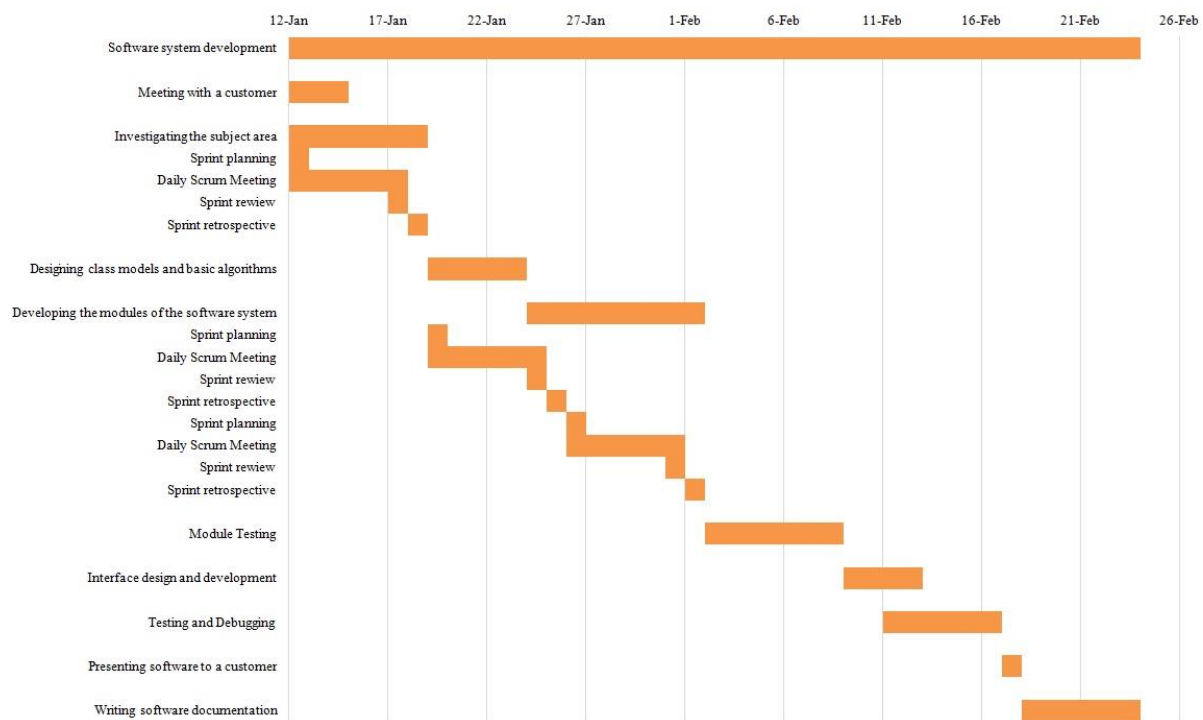


**Figure 4.** Critical path for a network diagram.

For the development of this software system we used Gantt charts to illustrate a project schedule [14]. This method is very popular in network planning, especially for small-scale projects. This diagram illustrates the project schedule. The lines are time-dependent. They show a time period allocated to a particular task of the project. The ends of the line correspond to task's start and finish dates. The key concept in the Gantt chart is a milestone - tool used to mark specific points along a project timeline.

The Gantt chart of our project is presented in figure 5. The stages of project development and their visual representation in the form of a band chart are presented.

One of the primary responsibilities of any manager is to provide comfortable work environment for the developers. This includes workplace, means of communication and necessary equipment, as well as implementation of project management strategy for coordinating task distribution in computer software development projects.



**Figure 5.** Gantt chart.

There are software development methodologies guided by values and principles established in Agile Manifesto. Scrum is one of the flexible approaches which have recently become popular. The

rules established for software project management allow developers to use the existing practices of coding, correcting requirements or making tactical changes. This methodology provides the ability to detect and eliminate deviations from the desirable result at earlier stages of the software product development. Iteration in Scrum when functional growth of software is created is sprint which is rigidly fixed in time (see figure 5). A sprint length of 1 week is appropriate as such a short sprint makes the process of software development more flexible, releases come earlier and more frequent, consumer responses arrive faster, and less time is wasted.

### 3. Project management tools

There are many software tools that are specialized to varying degrees in project management, as well as numerous general business software packages that support project management. To automate time-consuming tasks and run a large team of professionals, managers need the best project management software solutions for planning, tracking and monitoring software projects. These systems include Atlassian JIRA, Bugzilla, Redmine, GNATS, YouTrack, Trac and others. Most of them refer to issue tracking systems, which are sometimes used to manage software projects (e.g., JIRA). At the same time, some project management tools are also used for issue tracking (e.g., Redmine).

When programming, especially working remotely, software engineers have to control changes to source code. To track changes to software development projects version control systems are used. The most widely used modern version control system in the world today is Git. Public repositories on GitHub are often used to share open source software. It is the number-one hosting service for storing Git repositories and joint project development.

JIRA is a project and issue tracking software solution, developed by Atlassian. It was designed to improve code quality and speed development for software development teams. Its functionality can be expended by developing additional plug-ins, i.e. JIRA is a very flexible project management system. It provides bug tracking, issue tracking, and project management functions. JIRA was chosen from the tools options to develop the software system "Procurement Management".

Communication tools include instant messaging systems (messaging applications) which tend to facilitate connections between specified known users. While working on the project the Telegram was used by the developers. Slack is one more instant messaging system, which gains popularity in collaborative environment.

So, on the example of the software system "Procurement Management" we can conclude that an embodiment of a digital ecosystem is understood and treated in software engineering in accordance with the degree of interest and responsibility. There are no universal principles.

In fact two interacting digital platforms can build a separate ecosystem. The same platform can successfully become the part of different digital ecosystems due to high-quality of communication interfaces. Ultimately the ecosystems that provide open access on common and rational conditions will be the most competitive and actively developing. The example of such systems is Github with the simple idea to provide additional services based on decentralized Git technology and its users are able to retrieve their data at any time and refuse service easily.

It was D. Engelbart who considered the relations between people and software as a heterogeneous community in which there is evolution of all the agents involved [15]. In modern complex information systems there is co- evolution between people and tools. This is definitely a digital ecosystem. Engelbart noted that as soon as we begin to use new tools they change habitual conditions of our existence. "We expect that as tools are introduced and used, co-evolution will occur between the tools and the people using them. Thus, WYSIWYG systems eased the acceptance of computer systems by nontechnically oriented users; however, these systems produce a map of what you would see on paper as opposed to a hyper document with structural links evolving over time" [15].

Thus, in a broad sense the digital ecosystem is a biometaphor which suggests that modern organizations should be considered as the mixed communities and ecosystems in which people and digital agents interact in order to broaden human opportunities and expand noosphere with the use of digital storages and digital agents that help people to deal with these digital storages. For the



generalized digital ecosystem it is possible to allocate a number of essential concepts, such as: environment; communications; agent; multi-agent community; evolution.

In a narrow sense of software engineering the digital ecosystem is a distributed, adaptive, open social and technical system with the features of self-organization - scalability and stability in which there is a process of evolutionary development between the software product (or services) where the software agent is presented as an active agent and all intrasystem communications and communications with the external environment are implemented in topology of computer networks.

#### 4. Conclusions

Thus, we have considered some methods used to manage a software project on the example of developing the software system "Procurement Management". Although the problem of developing software systems is widely discussed in academic papers, some specific issues related to the integration of distributed digital ecosystems in software project management could be of interest for further research. This is the integration of network planning for the process of developing software system and Web 3.0 ecosystems. There is practically no research that is able to offer a systematic consideration of issue tracking systems and version control systems applicability in system engineering. The experience of using the Gantt chart in software engineering is discussed in some papers but it is incomplete. In this paper special attention is given to software tools used by project managers. The results of our research could be used for the development of multi-purpose software systems.

#### References

- [1] Bris P, Frantis M and Kolkova M 2015 Software quality control with the usage of ideal and TMMI models *MM Science Journal* **DECEMBER** 799–807
- [2] Fleming I 2015 Defining software quality characteristics to facilitate software quality control and software process improvement *Software Quality Assurance: In Large Scale and Complex Software-intensive Systems* 47–61
- [3] Evdokimov I V 2012 Information technology for accounting methodological support of the educational process *Issues of Social - Economic Development of Siberia* **4** 9–14
- [4] Evdokimov I V, Alalwan A R J, Timofeev N A and Nekhonoshin S R 2017 Internet of things in the context of software engineering economics and project cost management *On-line journal "Naukovedenie"* **9**
- [5] Evdokimov I V, Mikhalev A S, Timofeev N A and Baturin Yu A 2017 Forecasting the effectiveness of using virtual reality technologies in the educational process *Issues of Social - Economic Development of Siberia* **3** 129–135
- [6] Ritter M, Kuhr T, Hauth T, Gebard T, Kristof M and Pulvermacher C 2017 Software Quality Control at Belle II *Journal of Physics: Conference Series* **898** art. no. 072029
- [7] Solyman A M, Ibrahim O A and Elhag A A M 2016 Project management and software quality control method for small and medium enterprise *Proc. Int. Conf. on Computing, Control, Networking, Electronics and Embedded Systems Engineering* 123–128
- [8] Ratajczak-Ropel E and Skakovski A 2018 Project scheduling models *Studies in Systems, Decision and Control* **108** 25–32
- [9] Chernigovskiy A S, Tsarev R Yu and Kapulin D V 2017 Scheduling algorithms for automatic control systems for technological processes *Journal of Physics: Conference Series* **803** art. no. 012028
- [10] Chernigovskiy A S, Kapulin D V, Noskova E E, Yamskikh T N and Tsarev R Y 2017 Production scheduling with ant colony optimization *IOP Conference Series: Earth and Environmental Science* **87** art. no. 062002.
- [11] Hermans B and Leus R 2018 Scheduling Markovian PERT networks to maximize the net present value: New results *Operations Research Letters* **46** 240–244
- [12] Bhasin H and Gupta N 2018 Critical Path Problem for Scheduling Using Genetic Algorithm

*Advances in Intelligent Systems and Computing* **583** 15–24

- [13] Kelley J E and Walker M R 1959 Critical-path planning and scheduling *Proc. Eastern Joint Computer Conf. (IRE-AIEE-ACM)* 160-173
- [14] Wallace C 1922 *The Gantt Chart: A Working Tool of Management* (New York, NY: Ronald Press) p 190
- [15] Engelbart D, Lehtman H 1988 Working together *Byte* **13** 245-252